- PATTERN –

Patterns:

1. **Singleton pattern:** A pattern created to access the same object in various situations by creating one and only one object of a certain class.
   - You can create only one object per class and access only one object from all places.
2. **Strategy Pattern**: One function is defined and encapsulate each of them so that we can use them interchangeably.
   - By utilizing the strategy, the function (algorithm) can be changed independently of the client using the function (algorithm).

1. SINGLETON

- Has only one object (an instance of the class) at a time.
- Whatever modifications we do to any variable inside the class through any instance, affects the variable of the single instance created.
- A Singleton class has:
  I. A private constructor
  II. A static field containing its only instance
  III. A static factory method for obtaining the instance

Instance variables: Variables that are declared in a class, but outside a method, constructor or any block.

Singleton Class

```
public class SingletonClass {
        private int i = 10;
        private static SingletonClass INSTANCE;

        //LIKE SETTING A GETTER FOR THE VARIABLE INSTANCE
        public static SingletonClass getInstance() {
                if (INSTANCE == null) {
                        INSTANCE = new SingletonClass();
                }
                return INSTANCE;
        }
        private SingletonClass() { //CONSTRUCTOR
        }
        public int getI() {
                return i;
        }
        public void setI(int i) {

        this.i = i;
        }
}
```

if instance is NULL, set i=10, if not, return that value

First class

```
public class FirstClass {
    public FirstClass() {
        SingletonClass singletonObject = SingletonClass.getInstance();
        //CREATES A NEW OBJECT "singletonObject" with the value of INSTANCE
            System.out.println("Value of singleton object i: "+ singletonObject.getI());

            singletonObject.setI(999);
            System.out.println("After singleton class sets a new i value, i: " +
            singletonObject.getI());
        }
}
```

'i' of SingletonObject, of type SingletonClass is set to 999

Second class

```
public class SecondClass {
        public SecondClass()
                SingletonClass singletonObject = SingletonClass.getInstance();
                System.out.println("Running secondClass contructor");
                System.out.println("Value of singleton object i: " +
                                                        singletonObject.getI());
        }
```

TestMain

```
public class TestMain {
```

	`public static void main(String[] args) {`
	`FirstClass firstObj = new FirstClass();`
	`SecondClass secondObj = new SecondClass();`
	`SingletonClass singObj = SingletonClass.getInstance();`
	`System.out.println("Singleton object i's value at the 'main' function: " + singObj.getI());`
	`}`
	`}`

Result	**Value of singleton object i: 10** //Since INSTANCE is NULL, it returns the INSTANCE of the SingletonClass which is (i=10) **After singleton class sets a new i value, i: 999** **Running secondClass contructor** **Value of singleton object i: 999** **Singleton object i's value at the 'main' function: 999**

2. STRATEGY

SCHOOL	CAR	ROBOT
Person (parent class) Each person has their unique: - ID - Name - Belong **Each person can be categorised:** - Job - Income - Print	**Car (parent class)** Every car has their unique: - Can drive - Has features **Each Car can be categorised**: - Engine - Fuel - Km - Features (Override)	**Robot (parent class)** Every car has their unique: - Can walk - Can run **Each Robot can be categorised**: - Fly - Knife no knife) - Missile - Shape (override)
Student, staff & lecturer (child class): each extends 'Person' - Each sets their Job and Get according to their position	**Genesis, Sonata, Accent (child class):** each extends 'Car' Each sets their Engine, Fuel and Km according to their Car type	**Low, Standard and Super robot (child class):** each extends 'Robot' Each robot sets abilities in Flying, knife and missile.
Interfaces = What each person can be categorised as: IJob: - JobStudy (student) - JobMng (staff) - JobLec (lecturer) IGet: - Student pay (student) - Salary (staff & lecturer)	**Interfaces = What each car can be categorised as:** IEngine: - High engine - Mid engine - Low engine IFuel: - Diesel - Gasoline - Hybrid IKm: - Km 10 - Km 15 - Km 20	**Interfaces = What each car can be categorised as:** IFly: - Can fly - \cannot fly IKnife: - Lazer knife - Wood knife - No knife IMissile: - Has missile - No missile